NVM Express Technical Errata

Errata ID	**005**
Affected Spec Ver.	**NVM Express 1.0**
Corrected Spec Ver.	

Submission info

Name	Company	Date
Santosh	Samsung	3/16/2011
Kevin Marks	Dell	3/16/2011
Amber Huffman	Intel	3/16/2011

This erratum addresses editorial changes to the specification. Specifically:

- Figure 58 and 59 have minor typos that are corrected.
- Section 1 includes a substantial number of editorial changes for consistency and readability.
- Updates are made to Deallocate to reflect an appropriate reference to the ATA Trim functionality.

Description of the specification technical flaw

Modify byte 23:16 of Figure 58 as shown below:

23:16	**LBA:** This field indicates the first LBA that experienced the ~~erorr~~ error condition, if applicable.

Modify byte 0 of Figure 59 as shown below:

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>0</td><td><b>Critical Warning:</b> This field indicates critical warnings for the state of the controller. Each bit corresponds to a critical warning type; multiple bits may be set. If a bit is cleared to ‘0’, then that critical warning does not apply. Critical warnings may result in an asynchronous event notification to the host.
<table>
<tr><th>Bit</th><th>Definition</th></tr>
<tr><td>00</td><td>If set to ‘1’, then the available spare space has fallen below the threshold.</td></tr>
<tr><td>01</td><td>If set to ‘1’, then the temperature has exceeded a critical threshold.</td></tr>
<tr><td>02</td><td>If set to ‘1’, then the device reliability has been degraded due to significant media related errors or any internal error that degrades device reliability.</td></tr>
<tr><td>03</td><td>If set to ‘1’, then the media has been placed in read only mode.</td></tr>
<tr><td>04</td><td>If set to ‘1’, then the volatile memory backup device has failed. This field is only valid if the controller has a volatile memory backup solution.</td></tr>
<tr><td>~~15~~ 07:05</td><td>Reserved</td></tr>
</table>
</td></tr>
</table>

Modify the first part of section 1.4 as shown below:

1.4 Theory of Operation

Enhanced NVMHCI is a scalable host controller interface designed to address the needs of Enterprise and Client systems that utilize PCI Express based solid state drives. The interface provides an optimized command issue and completion path ~~beyond the mechanisms provided in AHCI and NVMHCI 1.0~~. It includes support for parallel operation by supporting up to 64K I/O Queues with up to 64K commands per I/O Queue ~~command queues within an I/O Queue~~. Additionally, support has been added for many Enterprise capabilities like end-to-end data protection (compatible with T10 DIF and SNIA DIX standards), enhanced error reporting, and virtualization.
The interface has the following key attributes:

- Does not require uncacheable / MMIO register reads in the command issue or completion path.
- A maximum of one MMIO register write is necessary in the command issue path.
- Support for up to 64K I/O queues, with each I/O queue supporting up to 64K commands.
- Priority associated with each I/O queue with ~~well defined~~ well-defined arbitration mechanism.
- All information to complete a 4KB read request is included in the 64B command itself, ensuring efficient small ~~random~~ I/O operation.
- Efficient and streamlined command set.
- Support for MSI/MSI-X and interrupt aggregation.
- Support for multiple namespaces.
- Efficient support for I/O virtualization architectures like SR-IOV.
- Robust error reporting and management capabilities.

This specification defines a streamlined set of registers whose functionality includes:

- Indication of controller capabilities
- Status for device failures (command status is processed via CQ directly)
- Admin Queue configuration (I/O Queue configuration processed via Admin commands)
- Doorbell registers for scalable number of Submission and Completion Queues

The capabilities that the controller supports are indicated in the Controller Capabilities (CAP) register and as part of the Controller and Namespace data structures returned ~~in~~ by the Identify command. The Identify Controller data structure indicates capabilities and settings that apply to the entire controller. The Identify Namespace data structure indicates capabilities and settings that are specific to a particular namespace.

Enhanced NVMHCI is based on a paired Submission and Completion Queue mechanism. Commands are placed by host software into ~~the~~ a Submission Queue. Completions are placed into ~~an~~ the associated Completion Queue by the controller. Multiple Submission Queues may utilize the same Completion Queue. ~~The ~~Submission and Completion Queues are allocated in host memory.

An Admin Submission and associated Completion Queue exist for the purpose of device management and control – (e.g., creation and deletion of I/O Submission and Completion Queues, aborting commands, etc.) Only commands that are part of the Admin Command Set may be issued to the Admin Submission Queue.

Modify the last three paragraphs of section 1.4 as shown below:

A Submission Queue (SQ) is a circular buffer with a fixed slot size that the host software uses to submit commands for execution by the controller. The host software updates the appropriate SQ Tail doorbell register when there are one to n new commands to execute. The previous SQ Tail value is overwritten in the controller when there is a new doorbell register write. The controller fetches SQ entries in order from the Submission Queue, however, it may then execute those commands in any order.

Each Submission Queue entry is a command. ~~The command is~~ Commands are 64 bytes in size. The physical memory locations in host memory to use for data transfers are specified using Physical Region Page (PRP) entries. Each command may include two PRP entries. If more than two PRP entries are necessary to describe the data buffer, then a pointer to a PRP List that describes a list of PRP entries is provided.

A Completion Queue (CQ) is a circular buffer with a fixed slot size used to post status for completed commands. A completed command is uniquely identified by a combination of the associated SQ identifier and command identifier that is assigned by host software. Multiple Submission Queues may be associated with a single Completion Queue. This feature may be used where a single worker thread processes all command completions via one Completion Queue even when those commands originated from multiple Submission Queues. The CQ Head pointer is updated by host software after it has processed completion queue entries indicating the last free CQ entry. A Phase (P) bit is defined in the completion queue entry to indicate whether an entry has been newly posted without consulting a register. This enables host software to determine whether the new entry was posted as part of the previous or current round of completion notifications. Specifically, each round through the Completion Queue ~~locations~~ entries, the controller inverts the Phase bit.

Modify the third to last paragraph of section 1.5 as shown below:

When a register bit is referred to in the document, the convention used is “Register Symbol.Field Symbol”. For example, the PCI command register parity error response enable bit is referred to by the name CMD.PEE. If the register field is an array of bits, the field ~~will be~~ is referred to as “Register Symbol.Field Symbol (array offset)”.

Modify section 1.6.1 as shown below:

1.6.1 Admin Queue

The Admin Queue is the Submission Queue and Completion Queue with identifier 0. The Admin Submission Queue and corresponding Admin Completion Queue are used to issue administrative commands and receive completions for those administrative commands, respectively.

~~Only the~~ The Admin Submission Queue is uniquely ~~may~~ associated ~~itself~~ with the Admin Completion Queue.

Modify section 1.6.3 as shown below:

1.6.3 arbitration mechanism

The method used to determine which Submission Queue is selected next to launch commands for execution by the controller. Three arbitration mechanisms are defined including round robin, weighted round robin with urgent priority class, and vendor specific. ~~A controller shall implement the round robin arbitration mechanism and may optionally implement the weighted round robin with urgent priority class arbitration mechanism and/or a vendor specific arbitration mechanism.~~ Refer to section 4.7.

Modify section 1.6.4 as shown below:

1.6.4 command completion

A command is completed when the controller has completed processing the command, has updated status information in the completion queue entry, and has posted the completion queue entry to the associated Completion Queue.

Modify section 1.6.8 as shown below:

1.6.8 firmware slot

A firmware slot is a location in the controller used to store a firmware image. The controller ~~shall~~ stores between one and seven firmware images. When downloading new firmware to the controller, ~~the~~ host software has the option of specifying which image ~~shall be~~ is replaced by indicating the firmware slot number.

Modify section 1.6.14 as shown below:

1.6.14 namespace

A namespace is a collection of logical blocks that range from 0 to the capacity of the namespace – 1. A namespace may or may not have a relationship to a Submission Queue; this relationship is determined by the host software implementation. The controller ~~shall~~ supports access to any valid namespace from any I/O Submission Queue.

Modify section 1.8 as shown below:

1.8 Conventions

A 0-based value is a numbering scheme for which the number 0h actually corresponds to a value of 1h and thus produces the pattern of 0h = 1h, 1h = 2h, 2h = 3h, etc. In this numbering scheme, there is not a method for specifying the value of 0h.

Some parameters are defined as a string of ASCII characters. ASCII data fields shall contain only code values 20h through 7Eh. For the string “Copyright”, the character “C” is the first byte, the character “o” is the second byte, etc. The string is left justified and shall be padded with spaces (ASCII character 20h) to the right if necessary.

Delete section 1.6.6 as shown below:

~~1.6.6 deallocate~~

~~The host may deallocate an LBA to indicate that the LBA is no longer in use by the host. Deallocate is similar to the Trim command in the ATA standard and the Unmap command in the SCSI standard. Refer to section 6.6.1.1.~~

Modify section 6.6.1.1 as shown below:

6.6.1.1   Deallocate

An LBA that has been deallocated using the Dataset Management command is no longer deallocated when the LBA is written. Read operations do not affect the deallocation status of an LBA. The value read from a deallocated LBA shall be deterministic; specifically, the value returned by subsequent reads of that LBA shall be the same until a write occurs to that LBA. The values read from a deallocated LBA shall be all zeros, all ones, or the last data written to the associated LBA.

Note: The operation of the Deallocate function is similar to the ATA DATA SET MANAGEMENT with Trim feature described in ACS-2 and SCSI UNMAP command described in SBC-3.

Modify section 1.11 as shown below:

1.11 References Under Development

ATA/ATAPI Command Set - 2 (ACS-2) [INCITS T13/2015-D]. Available from http://www.t13.org.

ISO/IEC 14776-323, SCSI Block Commands - 3 (SBC-3) [T10/1799-D]. Available from http://www.t10.org.

ISO/IEC 14776-454, SCSI Primary Commands - 4 (SPC-4) [T10/1731-D] Available from http://www.t10.org.

Trusted Computing Group Storage Interface Interactions Specification (SIIS). Available from http://www.trustedcomputinggroup.org.

Disposition log

3/16/2011	Erratum captured.
3/24/2011	Updated definitions to remove “shall” and added string justification.
3/29/2011	Modified Trim references.
5/10/2011	Erratum ratified.

Technical input submitted to the NVMHCI Workgroup is subject to the terms of the NVMHCI Contributor’s agreement.